Name: Mohit Kerkar Div: D15C Roll No: 23

Adv DevOps Exp 07

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Step 1:
Ensure Installations and path settings of the following softwares/services:

1. **Jenkins**
2. **Docker Desktop Engine**

Both these services help us create a sonarqube image and build a project within sonarqube using the pipeline architecture in jenkins.

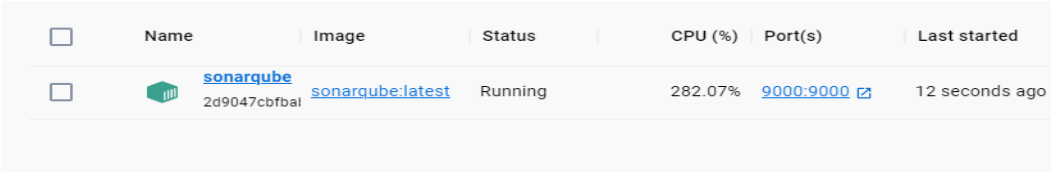
Run this command in powershell administrator mode: docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

This kickstarts the Sonarqube service/server at post **9000**. It also creates a Docker image within our Docker Desktop account. Make sure to keep your Docker Engine up and running

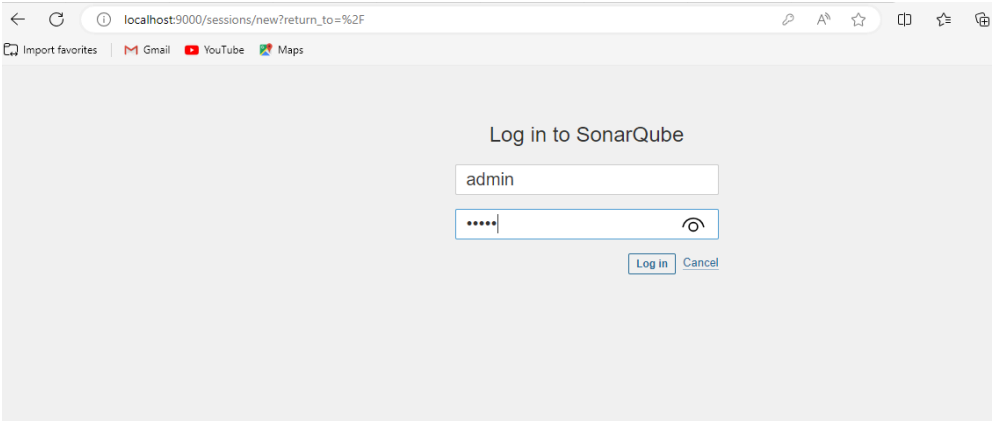
Powershell output:

```
PS C:\Users\INFT505-16> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:
latest
2d9047cbfbab55bd03ed72f7349feebf5d79434b23ff7eda329694bd24115349
PS C:\Users\INFT505-16>
```

Sonarqube started on Docker:

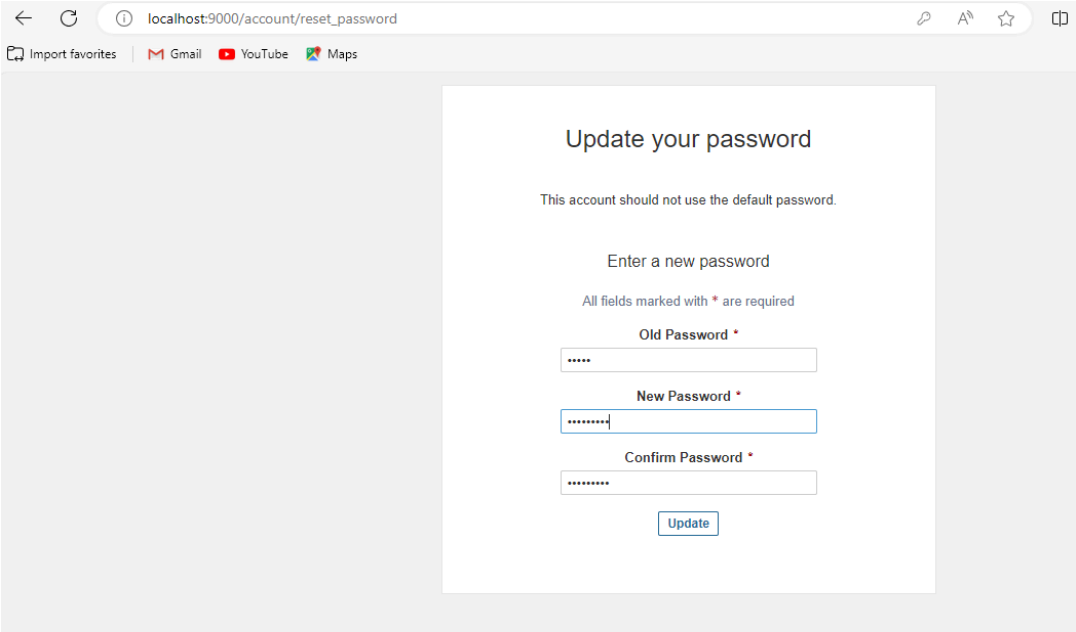


Go to localhost:9000. On fulfilling the desired action, we arrive on a page that asks us for our username and password.
The default credentials for Sonarqube server are (username,password) == **(admin,admin)**



Name: Mohit Kerkar Div: D15C Roll No: 23

Sonarqube asks us to update the default password and make a new password for this Sonarqube account. So i entered a new suitable password (keep it easy enough for one to remember)

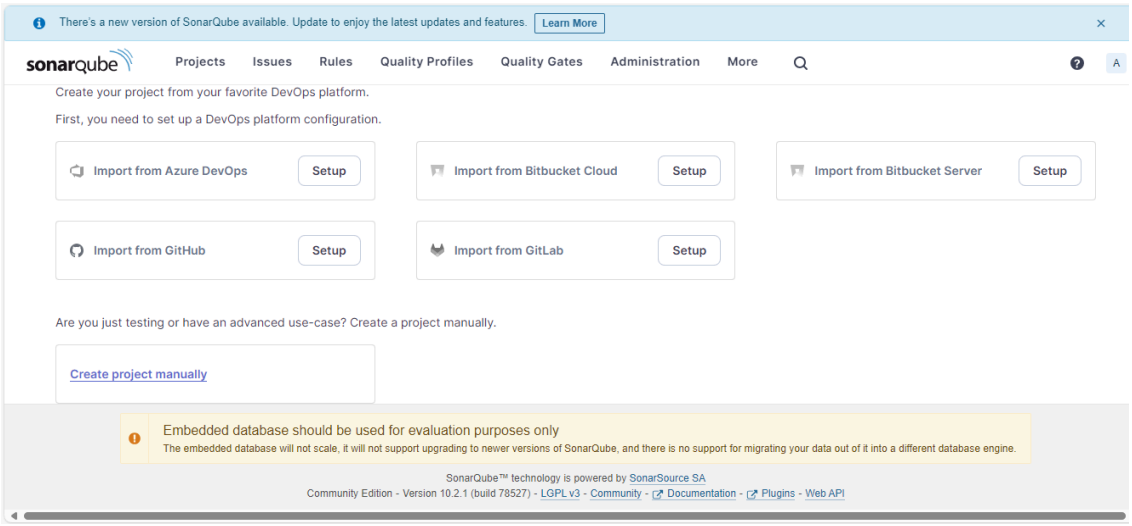


Step 2:

After logging in,

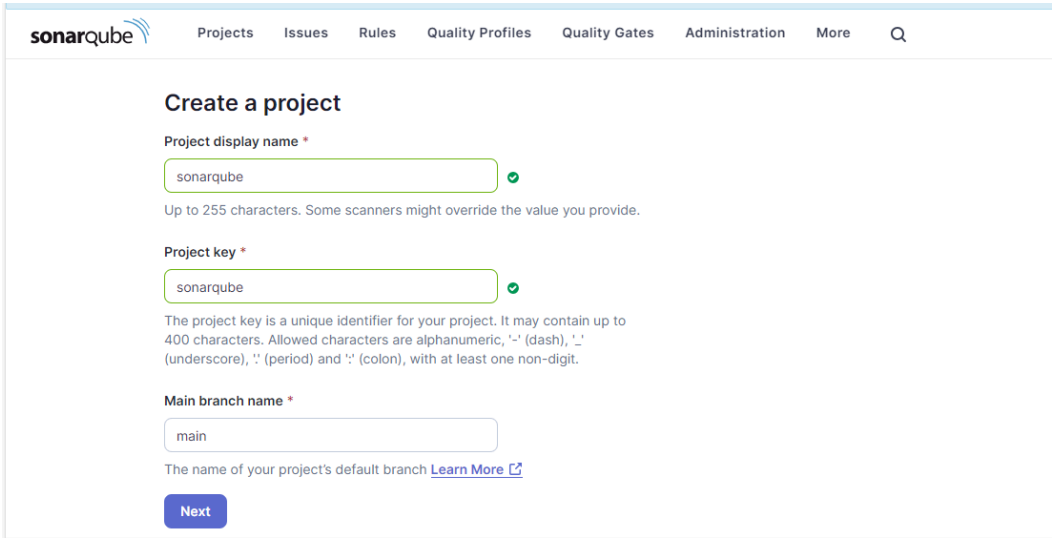
This is the page that we land on… This tells us that with the help of docker we have successfully created a Sonarqube image and that we have assigned a port number 9000 to it so as to use its services and build a Sonarqube project.

From here on, we are supposed to make a new project manually by selecting the ‘Create project manually’ option

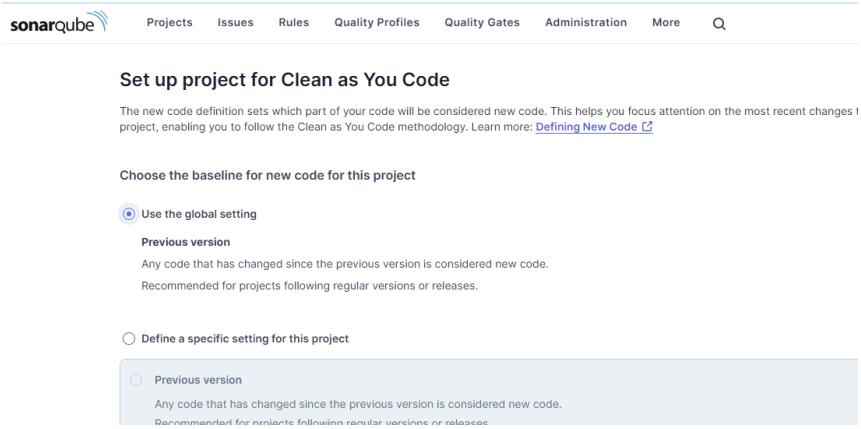


Name: Mohit Kerkar Div: D15C Roll No: 23

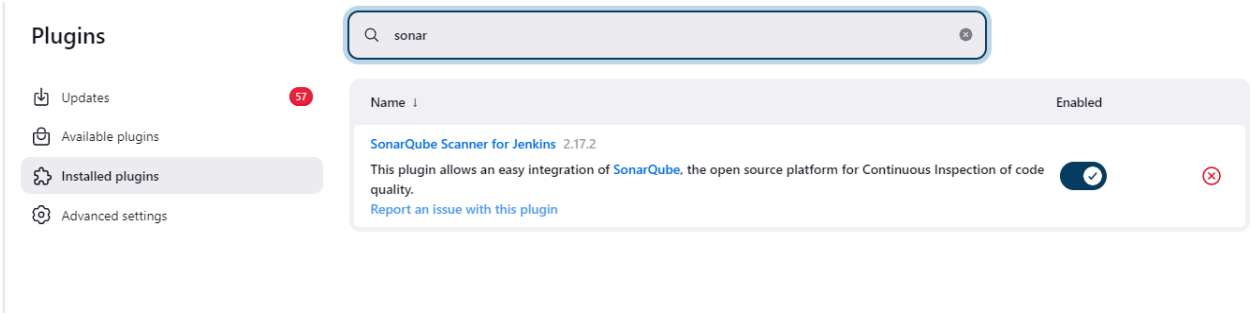
We are now supposed to enter the name for our Sonarqube
Lets keep it as ‘sonarqube’



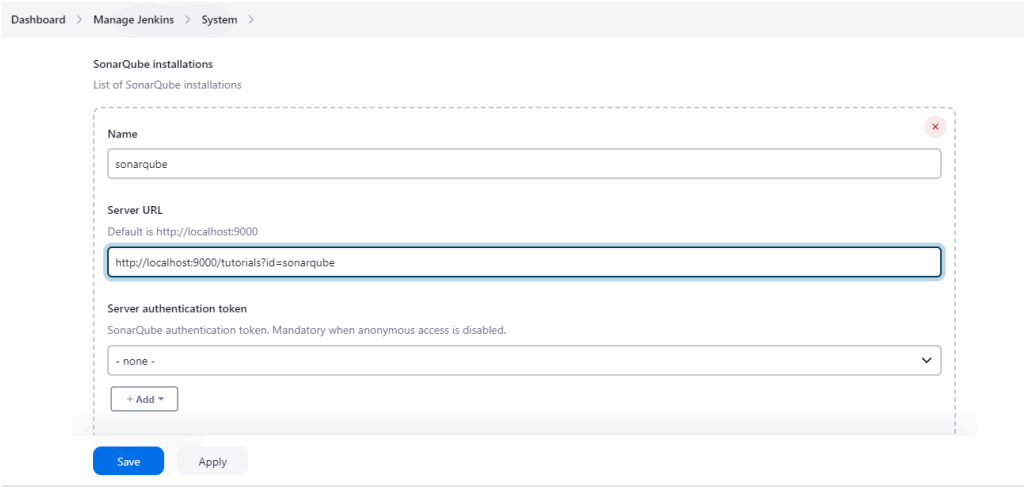
Next, we are asked for setting up our sonarqube project
We will select global settings for this project and finally create the project named ‘sonarqube’



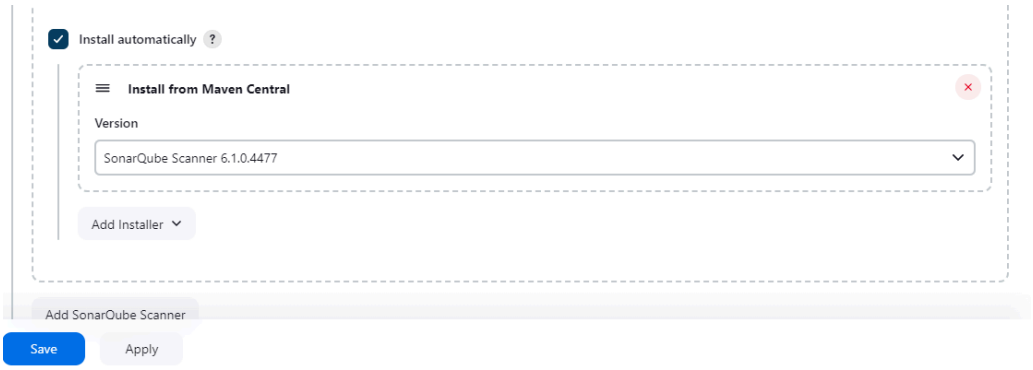
Step 3: Go to Jenkins Dashboard and download the plugin named **‘Sonarqube Scanner for Jenkins’**. This plugin will help us to build our Sonarqube project through a project created within jenkins. Make sure to restart Jenkins once installation is complete and Enable this plugin to use its services



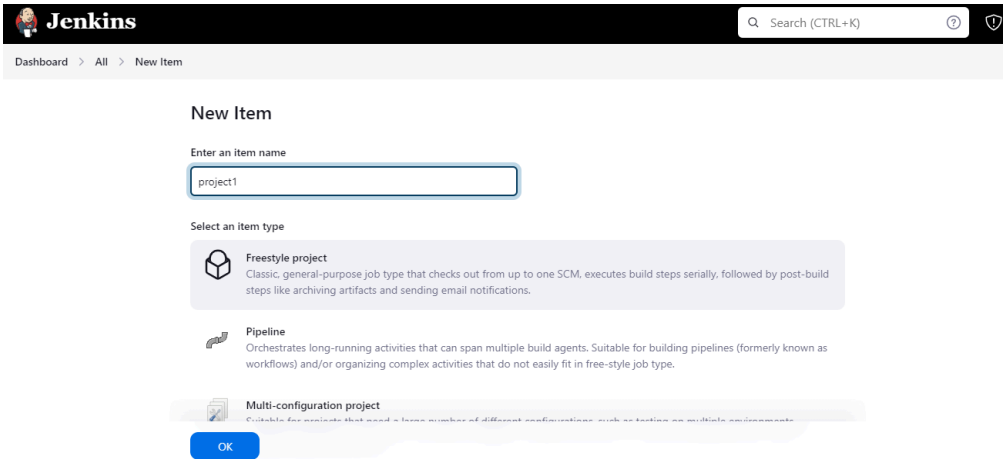
Go to jenkins system configurations, and navigate to Sonarqube installation
Enter the Sonarqube's project name, the URL and server authentication token if needed



Select 'install automatically' option for installation of the specified Sonnarqube Scanner version

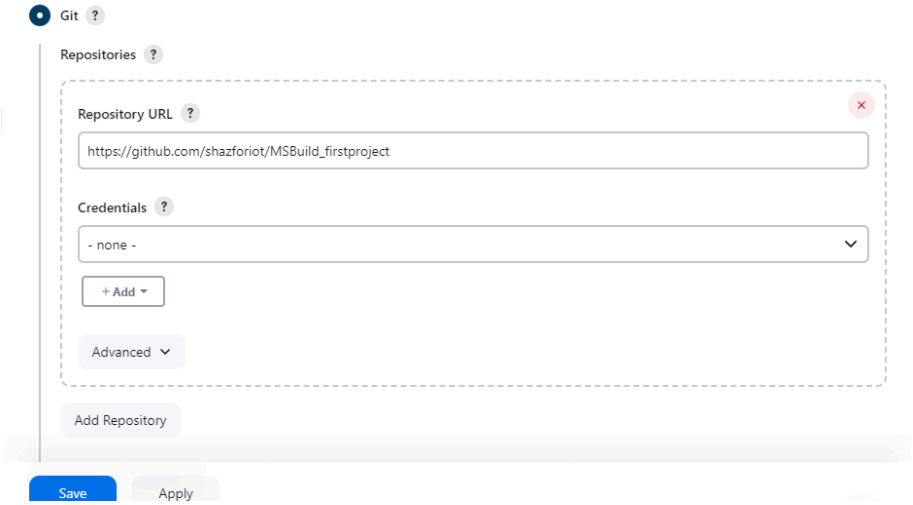


Step 4: Make a freestyle project in jenkins and give it a suitable name. This project is being made by us, so as to carry out our build of our Sonarqube project and perform static analysis of it based on whether the build is successful or not.

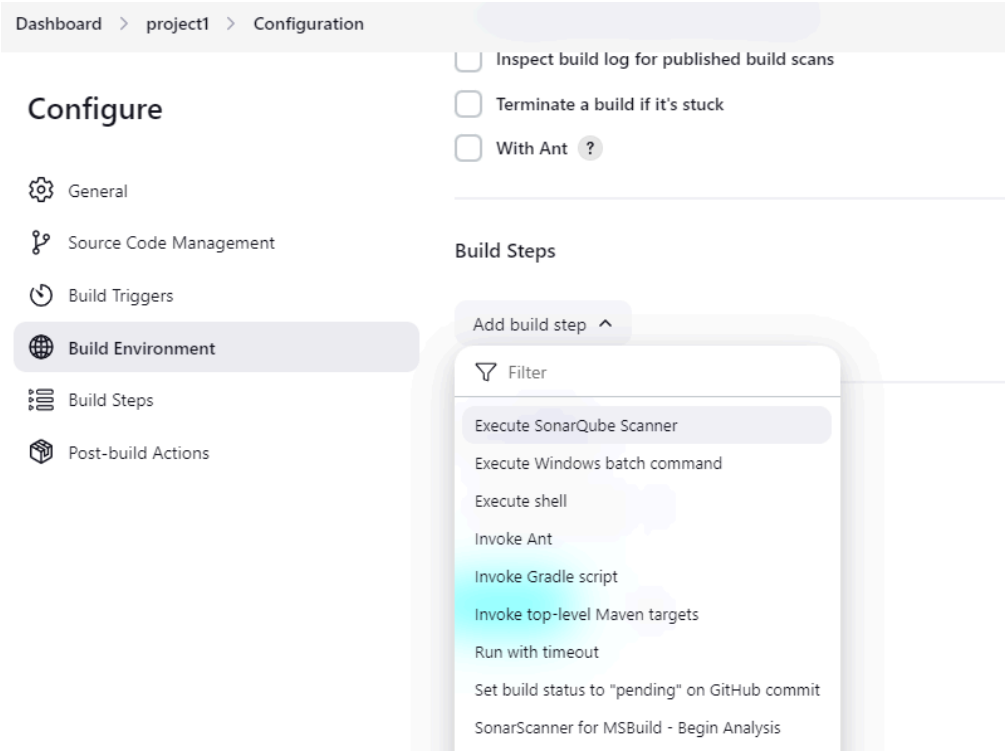


Name: Mohit Kerkar Div: D15C Roll No: 23

Choose this GitHub repository in Source Code Management.
https://github.com/shazforiot/MSBuild_firstproject.git
It is a sample hello-world project with no vulnerabilities and issues, just to test



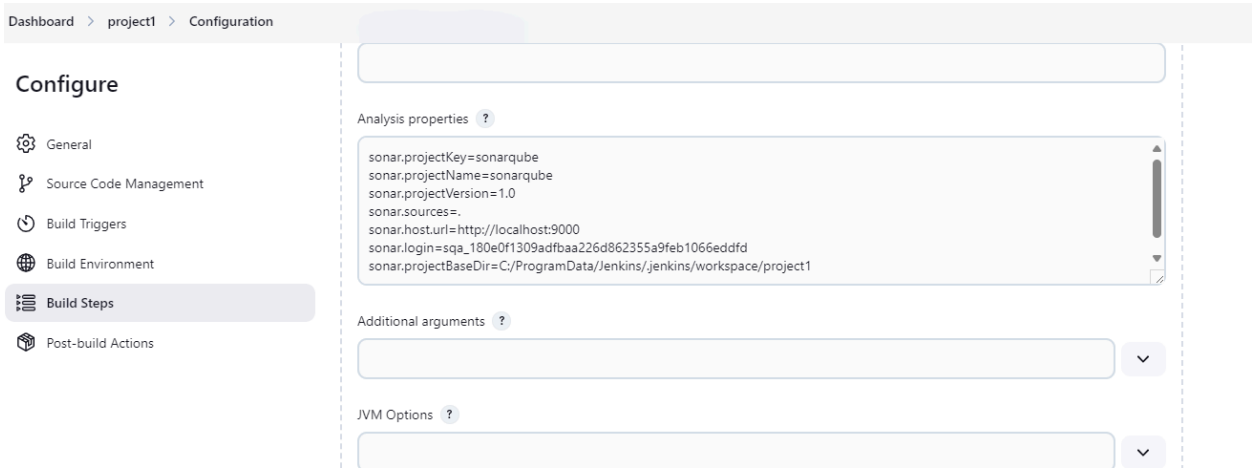
Next, under build steps, select ‘Execute Sonarqube Scanner’ option



Name: Mohit Kerkar Div: D15C Roll No: 23

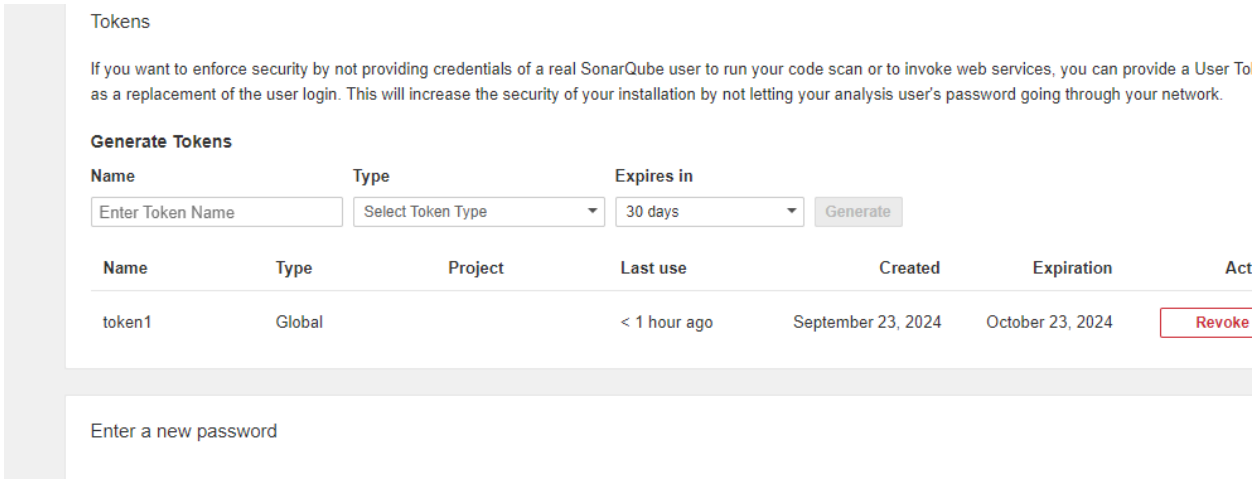
There is an option for entering ‘Analysis properties’ in which i entered the following credentials/parameters related to my Sonarqube account or the project that i have made:

sonar.projectKey=com.example.myproject
sonar.projectName=My Sample Project
sonar.projectVersion=1.0
sonar.sources=.
sonar.host.url=http://localhost:9000
sonar.login=sqa_180e0f1309adfbaa226d862355a9feb1066eddfd
sonar.projectBaseDir=C:/ProgramData/Jenkins/.jenkins/workspace/project1



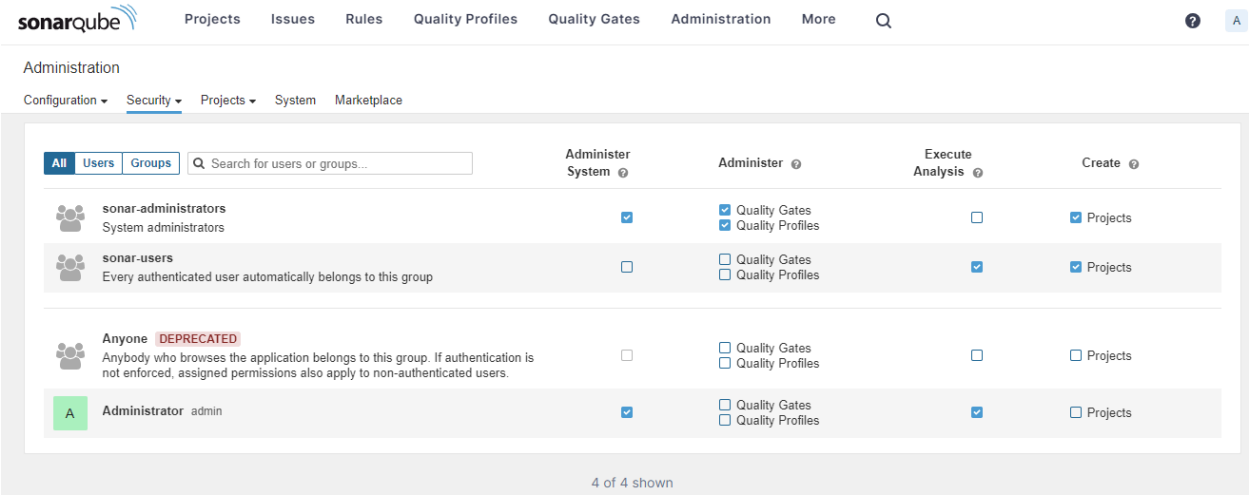
Step 5: In the analysis properties, in order to set the **sonar.login** parameter, we are supposed to navigate to our account details and choose the option of generating a global token. Create a new token…
Now, this assists us to identify our project uniquely without having to provide our credentials or in case our credentials fail to be identified

This token creation step is optional and is only to be used when our analysis properties consisting of our normal credentials fail

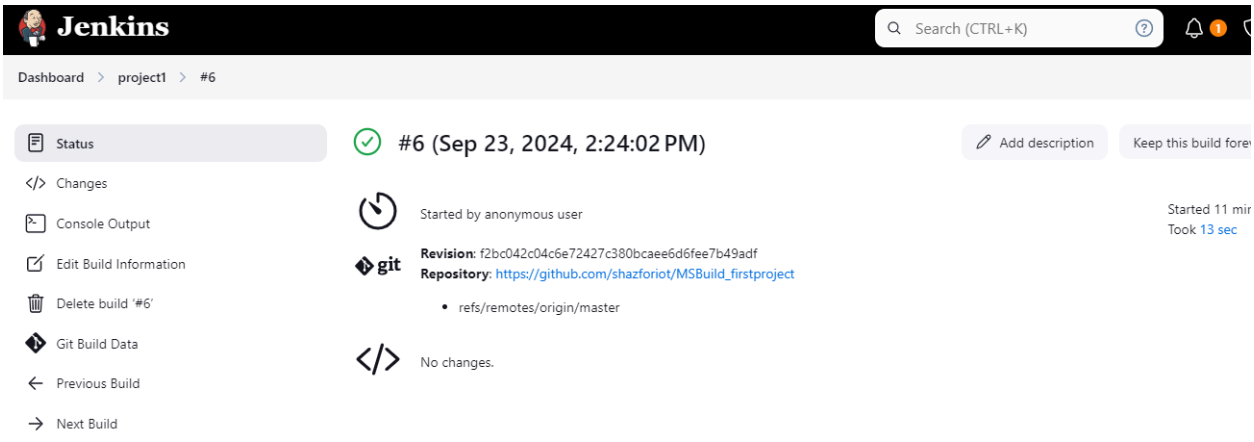


Name: Mohit Kerkar Div: D15C Roll No: 23

Now, navigate to security within administration and provide administrator system and execute analysis rights to the entity named admin.



After Building…the console output should indicate success
This indicates that the image that we created in docker successfully led us to creating a project within sonarqube, which in turn led us to building it over jenkins by uniquely identifying our sonarqube project. Build stage of our sonarqube project in jenkins involves static analysis for which we passed in the appropriate properties



Conclusion: The integration of Jenkins with SonarQube for static analysis using Docker provides a streamlined and automated approach to ensuring code quality. By successfully configuring the necessary tools—Docker for hosting SonarQube and Jenkins for building and analyzing the project—you can perform seamless static analysis using SonarQube's powerful scanning capabilities. This setup automated vulnerability detection, ensuring that potential issues are caught early. The use of GitHub as a repository for code allows for real-time scanning and continuous integration, ensuring that projects meet quality standards before moving forward in the development lifecycle.